# MYSQL容备与入坑

[MYSQL容备与入坑](https://blog.weiyigeek.top/2019/3-26-84.html)

##### update与delete 误删除恢复实战

描述:在实际运维中常常会发生将表中的数据进行update 以及 delete 等SQL语句未加WHERE条件时候，当日志记录没有开启时候，我们只能通过binlog文件进行查验导出恢复;

注意事项:binglog格式必须是ROW

#1.查看 binlog 文件列表  
mysql> show binary logs;  
#2.指定 inlog 文件查看  
mysql> show binlog events in 'mysql-bin.000008';

**1.update误操作恢复**

#环境假设  
update user set flag = 1 #此时没加上条件导致数据被该  
  
#恢复流程与步骤如下  
#1.如何数据库是在线上的话比较复杂，要先进行锁表，以免数据再次被污染；  
mysql> lock tables t1 read ;  
  
#2.查看当前写的二进制文件  
mysql> show master status;  
# +------------------+----------+--------------+------------------+  
# | File | Position | Binlog\_Do\_DB | Binlog\_Ignore\_DB |  
# +------------------+----------+--------------+------------------+  
# | mysql-bin.000024 | 1852 | | |  
  
  
#3.分析二进制日志，并且在其中找到相关记录  
#通过关键字  
mysqlbinlog --no-defaults -v -v --base64-output=DECODE-ROWS mysql-bin.000024 | grep -B 15 'zhuhai' #下十五条  
#通过时间段  
mysqlbinlog --no-defaults -v -v --base64-output=DECODE-ROWS binlog.000004 --start-datetime="2020-3-1 01:33:23" --stop-datetime="2020-3-1 01:40:22" | grep -C 5 --color "@4=3" > 1.txt  
#200301 1:33:23 server id 1 end\_log\_pos 1033911 CRC32 0x19eac976 Xid = 35805 COMMIT/\*!\*/;  
# at 1033911  
--  
### UPDATE `student`.`user`  
### WHERE  
### @1=49 /\* SHORTINT meta=0 nullable=0 is\_null=0 \*/  
### @2='杜邦能' /\* STRING(128) meta=65152 nullable=0 is\_null=0 \*/  
### @3=1 /\* TINYINT meta=0 nullable=1 is\_null=0 \*/  
### @4=3 /\* TINYINT meta=0 nullable=1 is\_null=0 \*/  
### @5=1583055203 /\* TIMESTAMP(0) meta=0 nullable=1 is\_null=0 \*/  
### SET  
### @1=49 /\* SHORTINT meta=0 nullable=0 is\_null=0 \*/  
### @2='杜邦能' /\* STRING(128) meta=65152 nullable=0 is\_null=0 \*/  
### @3=0 /\* TINYINT meta=0 nullable=1 is\_null=0 \*/ （误操作的部分）  
### @4=3 /\* TINYINT meta=0 nullable=1 is\_null=0 \*/  
### @5=1583055203 /\* TIMESTAMP(0) meta=0 nullable=1 is\_null=0 \*/  
  
#3.处理分析处理的二进制日志采用sed  
sed '/WHERE/{:a;N;/SET/!ba;s/\([^\n]\*\)\n\(.\*\)\n\(.\*\)/\3\n\2\n\1/}' t1.txt | sed -r '/WHERE/{:a;N;/@4/!ba;s/### @2.\*//g}' | sed 's/### //g;s/\/\\*.\*/,/g' | sed '/WHERE/{:a;N;/@1/!ba;s/,/;/g};s/#.\*//g;s/COMMIT,//g' | sed '/^$/d' > recover.sql   
grep -vE '@5' recover.sql > recover1.sql #排除@5的一行  
# 处理后的SQL  
# UPDATE `student`.`user`  
# SET  
# @1=49 ,  
# @2='杜邦能' ,  
# @3=1 ,  
# @4=3 ,  
# @5=1583055203 ,  
# WHERE  
# @1=49 ;  
# @5=1583055203 ,  
  
#4.将文件中的@1,@2,@3,@4替换为t1表中id，name，sex，address字段，并删除最后字段的","号  
sed -i 's/@1/id/g;s/@2/name/g;s/@3/flag/g;s/@4/grade/g;' recover1.sql  
sed -i -r 's/(grade=.\*),/\1/g' recover1.sql  
  
#5.登录MySQL数据库然后导入SQL语句恢复数据  
mysql> source recover.sql;

##### 2) delete 误操作恢复

#1.找到时间和POSITION位置节点,并且打印关键字之间的行  
mysqlbinlog --no-defaults --base64-output=decode-rows -v -v mysql-bin.000004 | sed -n '/### DELETE FROM db01.t1/,/COMMIT/p' > delete.txt  
### DELETE FROM db01.t1  
### WHERE  
### @1=1 /\* INT meta=0 nullable=0 is\_null=0 \*/  
### @2='daiiy' /\* STRING(60) meta=65084 nullable=0 is\_null=0 \*/  
### @3=2 /\* ENUM(1 byte) meta=63233 nullable=0 is\_null=0 \*/  
### @4='guangzhou' /\* VARSTRING(90) meta=90 nullable=0 is\_null=0 \*/  
  
#2.SQL语句转换  
cat delete.txt | sed -n '/###/p' | sed 's/### //g;s/\/\\*.\*/,/g;s/DELETE FROM/INSERT INTO/g;s/WHERE/SELECT/g;' | sed -r 's/(@4.\*),/\1;/g' | sed 's/@[1-9]=//g' > t1.sql  
  
#3.导入数据  
mysql> source t1.sql